Python 处理 Excel 数据之 Openpyxl

在前面《【曾贤志】从零基础开始用 Python 处理 Excel 数据》（链接地址：http://edu.51cto.com/sd/e9508），我们学习了 python 的相关知识。但是我们对 python 基础知识应用可能并不熟练，加上 xlrd、xlwt 两个处理 Excel 的库比较古老，功能也有局限性。因此我们推出了《Python 处理 Excel 数据之 Openpyxl》课程。有以下两个目的：

一是巩固前面学习的 python 基础知识。

二是学习一个功能更强大的处理 Excel 数据的库 openpyxl。

1.1 Openpyxl 库的安装使用

openpyxl 模块是一个读写 Excel 2010 文档的 Python 库，如果要处理更早格式的 Excel 文档，需要用到额外的库，openpyxl 是一个比较综合的工具，能够同时读取和修改 Excel 文档。其他很多的与 Excel 相关的项目基本只支持读或者写 Excel 一种功能。新建、读取、保存工作簿。Openpyxl 的安装和其它库一样。直接在 PyCharm 中安装即可。

Openpyxl 可以对 Excel 进入读、写操作，也可以进行单元格格式设置、图表、条件格式、数据透视表等设置。

由于本课程是为了巩固应用 python 基础知识，所以就只讲解对 openpyxl 对 Excel 的读写操作。

1.2 Excel 的新建、读取、保存

1.2.1 新建保存工作簿

新建：openpyxl.Workbook()，注意这个的 W 是大写的（本人吃过亏），可以设置 write_only=True/False 的读写方式，默认是可写。

保存：workbook.save('工作簿名.xlsx')

```
1-> from openpyxl import Workbook
2-> wb=Workbook()    #新建工作簿
3-> wb.save('我的工作簿.xlsx')  #保存工作簿
```

每个 workbook 创建后，默认会存在一个 worksheet。也可以自行创建新的 worksheet。

1.2.2 读取保存工作簿

读取工作簿：openpyxl.load_workbook('工作簿名.xlsx')，注意以下相关参数的设置。

read_only=False/True　False 表示可以读、写，True 表示只能读、不能写。

guess_types=False/True False 表示转换数据，True 表示不能转换数据。

data_only=False/True　False 表示序单元格的真实信息，True 表示只读取值。

```
1-> from openpyxl import load_workbook
2-> wb=load_workbook('成绩表-1.xlsx')    #读取工作簿
3-> wb.save('成绩表-2.xlsx')#保存工作簿
```

1.2.3 实例（批量建新工作表）

```
1-> from openpyxl import Workbook
2-> for m in range(1,13):
3->         wb = Workbook()    #新建工作簿
4->         wb.save('%d 月.xlsx'%m)#保存工作簿
```

1.3 工作表对象的获取方法

1.3.1 工作表获取方式

获取当前活动工作表的：workbook.active
以索引值方式获取工作表：workbook.worksheets[索引值]
以工作表名获取：workbook['工作表名']，注意，此表达方式没有没有成员提示。
循环工作表：workbook.worksheets
获取所有工作表名：workbook.sheetnames
获取指定工作表名：worksheet.title，可以返回工作表名称，也可以修改工作表名称，如worksheet.title='工作表名'

1.3.2 实例（批量修改工作表名）

```
1-> import openpyxl
2-> wb=openpyxl.load_workbook('各年业绩表.xlsx')
3-> for sh in wb.worksheets:
4->         sh.title=sh.title+'-芝华公司'
5-> wb.save('各年业绩表（修改后）.xlsx')
```

1.4 工作表的新建、复制、删除

1.4.1 新建工作表

可以在新建的工作簿中新建工作表（在新建工作簿时，会默认新建一个工作表）。也可在已经存在的工作簿中新建工作表。

新建工作表时的默认工作表名：workbook.create_sheet()，默认工作表名为 Sheet1、Sheet2、Sheet3……

新建工作表自定义工作表名：workbook.create_sheet('工作表名',指定位置)，如果不指定位置则默认将新建的工作表放置在最后。

1.4.2 复制工作表

workbook.copy_worksheet(工作表)

1.4.3 删除工作表

workbook.remove(工作表)

1.5 关于工作表的实例应用

1.5.1 实例应用（批量新建工作表）

```
1-> import openpyxl
2-> wb = openpyxl.Workbook()    #新建工作簿
3-> for m in range(1,13):
4->         wb.create_sheet('%d 月'%m)#新建月份工作表
5-> wb.remove(wb['Sheet']) #删除指定工作表
6-> wb.save('2019 年计划表.xlsx')      #保存工作簿。
```

1.5.2 实例应用（删除不符合条件的工作表）

```
1-> import openpyxl
2-> wb=openpyxl.load_workbook('2018 年.xlsx')      #读取工作簿
3-> for sh in wb:   #循环工作簿中的工作表
4->         if sh.title.split('-')[0]!='北京':  #判断工作表是否不等于北京
5->                 wb.remove(sh)      #删除工作表
6-> wb.save('北京.xlsx')      #保存工作簿
```

1.5.3 实例应用（批量复制工作表）

```
1-> import openpyxl
2-> wb=openpyxl.load_workbook('模板.xlsx')
3-> for m in range(1,13):
4->         wb.copy_worksheet(wb['demo']).title='%d 月'%m
5-> wb.remove(wb['demo'])
6-> wb.save('2018 年各月表格.xlsx')
```

1.6 单元格信息获取

1.6.1 单元格数据获取

A1 表示法：**工作表['A1']**，R1C1 表示法：**工作表.cell(行号,列号)**

1.6.2 实例应用（汇总各表各单元格数据）

```
1-> import openpyxl
2-> wb = openpyxl.load_workbook('各年业绩表.xlsx')
3-> print(sum([s['b14'].value for s in wb]))
4-> print(sum([s.cell(14,2).value for s in wb]))
```

1.7 单元格区域信息获取

1.7.1 单元格区域数据获取

1.工作表['起始单元格': '终止单元格']或工作表['起始单元格: 终止单元格']，如 ws['A1': 'F3']或 ws['A1: F3']。此方法是按行读取的数据。
2.工作表['起始行号': '结束行号']或者工作表['起始行号: 结束行号']，如 ws['1': '3']或 ws['1: 3']。此方法是按行读取的数据。
3.工作表['起始列号': '结束列号']或者工作表['起始列号: 结束列号']，如 ws['A': 'F'] 或 ws['A: F']。此方法是按列读取的数据。
4.获取（按行）指定工作表所有已用数据：list(workbook.worksheets[索引值].values)

1.7.2 实例应用

按行求和（方法 1）

```
1-> import openpyxl
2-> wb = openpyxl.load_workbook('test.xlsx')
3-> ws=wb['成绩表']
4-> # rng=ws['2:71']
5-> rng=ws['A2':'E71']
6-> print(['%s:%d 分'%(rn[0].value,sum([r.value for r in rn][1:])) for rn in rng])
```

按行求和（方法 2）

```
1-> import openpyxl
2-> wb=openpyxl.load_workbook('test.xlsx')
3-> ws=wb.active
4-> for x in list(ws.values)[1:]:
5->         print([x[0],sum(x[1:])])
```

按列统计平均值

```
1-> import openpyxl
2-> wb=openpyxl.load_workbook('test.xlsx')
3-> ws=wb.active
4-> for x in list(zip(*list(ws.values)))[1:]:
5->         print([x[0],float('%.2f'%(sum(x[1:])/len(x)-1))])
```

1.8 行列信息获取

1.8.1 行列信息获取

按行获取工作表使用区域数据：worksheet.rows
按列获取工作表使用区域数据：worksheet.columns
获取工作表中最小行号：worksheet.min_row
获取工作表中最小列号：worksheet.min_column
获取工作表中最大行号：worksheet.max_row
获取工作表中最大列号：worksheet.max_column
获取单元格的行号：cell.row
获取单元格的列号：cell.column

iter 方法获取指定区域：

1.按行获取指定工作表单元格区域：worksheet.iter_rows(……)
2.按列获取指定工作表单元格区域：worksheet.iter_cols(……)
可以通过 min_row、min_col、max_col、max_row 这几个参数进行单元格区域的控制

1.8.2 实例应用

按行求和

```
1-> import openpyxl
2-> wb=openpyxl.load_workbook('test.xlsx')
3-> ws=wb.active
4-> for r in [row for row in ws.rows][1:]:
5->     l=[v.value for v in r]
6->     print([l[0],sum(l[1:])])
```

按列求最大值

```
1-> import openpyxl
2-> wb=openpyxl.load_workbook('test.xlsx')
3-> ws=wb.active
4-> for c in [col for col in ws.columns][1:]:
5->     l=[v.value for v in c]
6->     print([l[0],max(l[1:])])
```

按行求和

```
1-> import openpyxl
2-> wb=openpyxl.load_workbook('test.xlsx')
3-> ws=wb.active
4-> subtotal=[sum([v.value for v in row]) for row in
    ws.iter_rows(min_row=2,min_col=2)]
5-> name=[v.value for v in ws['a']][1:]
6-> print(list(zip(name,subtotal)))
```

按列求最大值

```
1-> import openpyxl
2-> wb=openpyxl.load_workbook('test.xlsx')
3-> ws=wb.active
4-> subtotal=[max([v.value for v in col]) for col in ws.iter_cols(min_row=2,min_col=2)]
5-> name=[v.value for v in ws['1']][1:]
6-> print(list(zip(name,subtotal)))
```

动态获取单元格区域并汇总

```
1-> import openpyxl
2-> wb=openpyxl.load_workbook('demo.xlsx')
3-> ws=wb.active
4-> minr=ws.min_row
5-> minc=ws.min_column
6-> maxr=ws.max_row
7-> maxc=ws.max_column
8-> rngs=ws.iter_rows(min_row=minr+1,min_col=minc+2,max_row=maxr-1,max_col=
    maxc-1)
9-> subtotal=[min([v.value for v in row]) for row in rngs]
10-> col=[v for v in
    ws.iter_cols(min_row=minr+1,min_col=minc+1,max_row=maxr-1,max_col=minc+1)
    ]
11-> chanping=[[v.value for v in r] for r in col][0]
12-> print(list(zip(chanping,subtotal)))
```

1.9 单元格的写入

1.9.1 单元格与区域数据写入

A1 表示法：**工作表['A1']=值**，R1C1 表示法：**工作表.cell(行号,列号,值)**

1.9.2 实例应用（九九乘法表）

```
import openpyxl
wb=openpyxl.load_workbook('demo.xlsx')
ws=wb.active
for x in range(1,10):
    for y in range(1,x+1):
        ws.cell(x,y,'%d×%d=%d'%(y,x,x*y))#方法1
        ws.cell(x,y).value='%d×%d=%d'%(y,x,x*y)#方法2
wb.save('demo.xlsx')
```

1.10 批量写入数据

1.10.1 按行写入数据

在最后一行写入数据：工作表.append(列表)

1.10.2 实例应用（九九乘法表）

```
import openpyxl
wb=openpyxl.load_workbook('demo.xlsx')
ws=wb.active
l=[['%d×%d=%d'%(y,x,x*y) for y in range(1,10) if y<=x] for x in range(1,10)]
for r in l:
    ws.append(r)
wb.save('demo.xlsx')
```

1.11 循环方式批量写入数据

1.11.1 循环获取单元格对象，再写入

之前我们可以通过组合单元格来获取或者写入数据，但还有一种方法，就是直接循环单元格区域来写入数据。与循环读取的表示方式基本相同，只是多了一个赋值。

1.11.2 实例应用（大于等于 90 分为优秀）

```
1-> import openpyxl
2-> wb=openpyxl.load_workbook('demo.xlsx')
3-> ws=wb.active
4-> rngs=ws.iter_rows(min_row=2,min_col=2)
5-> for row in rngs:
6->         for c in row:
7->                 if c.value>=90:
8->                         c.value='%d(%s)'%(c.value,'优秀')
9-> wb.save('demo1.xlsx')
```

1.11.3 实例应用（每个人的总分大于等于 300 为优秀）

```
1.  import openpyxl
2.  wb=openpyxl.load_workbook('demo.xlsx')
3.  ws=wb.active
4.  rngs=ws.iter_rows(min_row=2,min_col=2)
5.  for row in rngs:
6.                  sm=sum([c.value for c in row][0:4])
7.                  if sm>=300:
8.                          row[-1].value='优秀'
9.  wb.save('demo2.xlsx')
```

1.12 工作表行、列的插入与删除

1.12.1 行列的插入与删除

插入列：worksheet.insert_cols(位置,列数)，其中位置是指在工作表的第几列前插入多少列。
插入行：worksheet.insert_rows(位置,行数)，其中位置是指在工作表的第几行前插入多少行。
删除列：worksheet.delete_cols(位置,列数)，从指定位置开始向后删除指定的列数。
删除行：worksheet.delete_rows(位置,行数)，从指定位置开始向下删除指定的行数。

1.12.2 实例应用（筛选小于 300 的记录）

```
12-> import openpyxl
13-> wb=openpyxl.load_workbook('成绩表.xlsx')
14-> ws=wb.active
15-> for r in range(ws.max_row,1,-1):
16->     s=sum([v.value for v in ws[r][1:]])
17->     if s>=300:
18->         ws.delete_rows(r)
19-> wb.save('demo999.xlsx')
```

1.13 实例应用（求和结果写入单元格）

```
1-> import openpyxl
2-> wb=openpyxl.load_workbook('test.xlsx')
3-> ws=wb['成绩表']
4-> rng=ws[str(ws.min_row+1):str(ws.max_row)]#动态获取单元格区域
5-> rngs=[[rn[0].value,sum([r.value for r in rn][1:])] for rn in rng]#汇总处理
6-> ws1=wb.create_sheet('结果')#新建工作表
7-> ws1.append(['姓名','总分数'])#写入表头
8-> for line in rngs:
9->     ws1.append(line)#循环写入求和分数
10-> wb.save('test1.xlsx')#保存工作簿
```

1.14 实例应用（筛选成绩总分大于等于 300 分的记录）

```
1-> import openpyxl
2-> wb=openpyxl.load_workbook('test.xlsx')
3-> ws=wb['成绩表'];nws=wb.create_sheet('结果')
4-> rng=list(ws.rows)[1:];nws.append([v.value for v in ws['1':'1']]+['总分'])
5-> for l in rng:
6->     ll=[v.value for v in l]
7->     if sum(ll[1:])>=300:
8->         nws.append(ll+[sum(ll[1:])])
9-> wb.save('test2.xlsx')
```

1.15 实例应用（工资条制作）

```
import openpyxl
wb=openpyxl.load_workbook('工资表.xlsx',data_only=True)
ws=wb.active
for r in range(ws.max_row,2,-1):
    ws.insert_rows(r)
    for c in range(1,8):
        ws.cell(r,c,ws.cell(1,c).value)
wb.save('工资表结果.xlsx')
```

1.16 实例应用（多工作表合并到单工作表）

```
import openpyxl
wb=openpyxl.load_workbook('各年业绩表.xlsx')#读取工作簿
nwb=openpyxl.Workbook()#新建工作簿
nws=nwb.active#获取活动工作表
nws.append(['年份','月份','金额'])#写入标题
for sh in wb:
    ll=[[sh.title] + [v.value for v in l] for l in sh.rows][1:-1]#合并各表数据
    for l in ll:
        nws.append(l)#写入到新表
nwb.save('合并.xlsx')#保存工作表
```

1.17 实例应用（单工作表拆分到多工作表）

```
import openpyxl
wb=openpyxl.load_workbook('各班成绩表.xlsx')#读取工作簿
ws=wb.active#读取活动工作表
rngs=ws.iter_rows(min_row=2)#获取工作表中的数据
d={}#创建空字典
for r in rngs:
    l=[v.value for v in r]
    if l[0] in d.keys():#将每行数据写入到字典中对应的班级
        d[l[0]]+=[l]
    else:
        d[l[0]]=[l]
nwb=openpyxl.Workbook()#新建工作簿
for k,v in sorted(d.items()):
    nws=nwb.create_sheet(k)#将字典中的键名做为班级名
    nws.append(['班级','姓名','分数'])#写入每个工作表的标题
    for r in v:
        nws.append(r)#将每个班的记录写入对应的工作表
nwb.remove(nwb['Sheet'])#删除默认创建的工作表
nwb.save('拆分到工作表.xlsx')#保存工作簿
```

1.18 实例应用（单工作簿拆分到多工作簿中（单表中））

```
import openpyxl
wb=openpyxl.load_workbook('工资表.xlsx',data_only=True)
rngs=wb.active.iter_rows(min_row=2)
d={}
for row in rngs:
    l=[v.value for v in row]
    if l[2] in d.keys():
        d[l[2]]+=[l]
    else:
        d.update({l[2]:[l]})
for k,v in d.items():
    nwb=openpyxl.Workbook()
    nws=nwb.active;nws.title=k
    nws.append(['工号','姓名','应发工资','扣款','奖金','实发工资'])
    for r in v:
        del r[2]
        nws.append(r)
    nwb.save('各部门工资表\\'+k+'.xlsx')
```

1.19 实例应用（单工作簿拆分到多工作簿中（多表中））

```
import openpyxl
wb=openpyxl.load_workbook('工资表.xlsx',data_only=True)
ws=wb.active
rngs=ws.iter_rows(min_row=2)
d={}
for row in rngs:
    ll=[l.value for l in row]
    if ll[2] in d.keys() and ll[3] in d[ll[2]].keys():
        d[ll[2]][ll[3]]+=[ll]
    else:
        if not ll[2] in d.keys():d[ll[2]]={}
        d[ll[2]][ll[3]]=[ll]
for k,v in d.items():
    nwb=openpyxl.Workbook()
    for m,n in v.items():
        nws=nwb.create_sheet(m);nws.append([v.value for v in ws['1']])
        for f in n:
            nws.append(f)
    nwb.remove(nwb['Sheet'])
    nwb.save('拆分结果\\'+k+'.xlsx')
```

1.20 实例应用（二维表转一维表）

```
import openpyxl
wb=openpyxl.load_workbook('业绩表.xlsx')
ws=wb.active
if '转换表' in wb.sheetnames:
    wb.remove(wb['转换表'])
nws=wb.create_sheet('转换表')
nws.append(['姓名','月份','金额'])
for name,row in zip(ws['a'][1:],ws.iter_rows(min_col=2,min_row=2)):
    for x,y in zip(ws['1'][1:],row):
        nws.append([name.value,x.value,y.value])
wb.save('业绩表.xlsx')
```

1.21 实例应用（一维转二维）

```
import openpyxl
wb=openpyxl.load_workbook('业绩表.xlsx')
ws=wb.active
if not '二维表' in wb.sheetnames:
    nws=wb.create_sheet('二维表')
    rngs=[[r.value for r in row] for row in ws.iter_rows(min_row=2)]
    mm=list({name.value:'' for name in ws['b'][1:]}.keys())
    nws.append(['姓名']+mm)
    for x in {name.value:'' for name in ws['a'][1:]}.keys():
        l=[(x,y) for y in mm]
        nws.append([x]+[list(filter(lambda x:x[0]==s[0] and x[1]==s[1],rngs))[0][2] for s in l])
wb.save('业绩表.xlsx')
```

1.22 实例应用（将入库单据数据写入工作表）

```
import openpyxl
wb=openpyxl.load_workbook('入库单.xlsx',data_only=True)
wb1=openpyxl.load_workbook('数据库.xlsx')
ws=wb.active
l=list(ws.values)
t=(l[2][1],l[2][3],l[2][5])
if not t[2] in [v.value for v in wb1.active['i']]:
    for v in l[5:]:
        if not None in v:
            wb1.worksheets[0].append(v+t)
    wb1.save('数据库.xlsx')
    print('保存成功！')
else:
    print('已保存')
```